1990-03

A pipelined implementation of notch filters using Genesil silicon compiler

Kung, Chih-fu

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A PIPELINED IMPLEMENTATION OF
NOTCH FILTERS USING GENESIL
SILICON COMPILER

by

Kung, Chih-Fu

March 1990

Thesis Advisor: Chyan Yang

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED

1b RESTRICTIVE MARKINGS:

2a SECURITY CLASSIFICATION AUTHORITY:

2b DECLASSIFICATION / DOWNGRADING SCHEDULE:

3 DISTRIBUTION / AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited

4 PERFORMING ORGANIZATION REPORT NUMBER(S):

5 MONITORING ORGANIZATION REPORT NUMBER(S):

6a NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School

6b OFFICE SYMBOL *(If applicable)***:** AS

7a NAME OF MONITORING ORGANIZATION: Naval Postgraduate School

6c ADDRESS *(City, State, and ZIP Code)***:** Monterey, CA 93943-5000

7b ADDRESS *(City, State, and ZIP Code)***:** Monterey, CA 93943-5000

8a NAME OF FUNDING SPONSORING ORGANIZATION:

8b OFFICE SYMBOL *(If applicable)***:**

9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER:

8c ADDRESS *(City, State, and ZIP Code)***:**

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE *(Include Security Classification)***:** A PIPELINED IMPLEMENTATION OF NOTCH FILTERS USING GENESIL SILICON COMPILER

12 PERSONAL AUTHOR(S): KUNG, Chih-Fu

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM ______ TO ______	1990 March	54

16 SUPPLEMENTARY NOTATION: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government.

17 COSATI CODES

FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)***:** IIR; Notch Filter; Pipeline; CMOS; Silicon Compiler, One's Complement; Two's Complement

19 ABSTRACT *(Continue on reverse if necessary and identify by block number)*

To implement an IIR notch filter is theoretically feasible but not technically verified or validated. Two methods often used to speed up a computation are multiprocessing and pipelining. In designing a notch filter for the pipelining technique is the natural choice to speed up its processing speed. To have a rapid prototype design we may employ the silicon compiler techniques and explore numerous design variations before sending for fabrication.

This paper will report the alternative pipelined design of IIR notch filters. We will present the problem, explain the methodologies used in our investigation, analyze the results, and discuss the findings. We first summarize various fixed-point designed for the pipeline building component, the multiplier-added pair. We then present the design considerations

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21 ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED

22a NAME OF RESPONSIBLE INDIVIDUAL: YANG, Chyan

22b TELEPHONE *(Include Area Code)***:** 408-646-2266

22c OFFICE SYMBOL: EC/Ya

DD Form 1473, JUN 86 — *Previous editions are obsolete* — SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-0 6603 UNCLASSIFIED

19. cont.
about the system integration. Various parameters are investigated in our research: pipelined stages, timing, silicon area. Additionally, the experiences and difficulties of using timing verifiers that are built in the silicon compiler will be discussed as well.

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Approved for public release; distribution is unlimited.

A Pipelined Implementation of Notch Filters Using Genesil Silicon Compiler

by

Kung, Chih-fu
Lieutenant Commander, Taiwan Navy
B.S., Chinese Naval Academy on Taiwan, 1981

Submitted in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE IN TELECOMMUNICATION SYSTEMS MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
March 1990

Author: [redacted]
Kung, Chih-fu

Approved by: [redacted]
Chyan Yang, Thesis Advisor

[redacted]
Herschel H. Loomis, Jr. Thesis Co-advisor

[redacted]
Dan C. Boger, Second Reader

[redacted]
David R. Whipple
Chairman, Engineering Acoustics Academic Committee

ABSTRACT

To implement an IIR notch filter is theoretically feasible but not technically verified or validated. Two methods often used to speed up a computation are multiprocessing and pipelining. In designing a notch filter the pipelining technique is the natural choice to speed up its processing speed. To have a rapid prototype design we may employ the silicon compiler techniques and explore numerous design variations before sending for fabrication.

This paper will report the alternative pipelined design of IIR notch filters. We will present the problem, explain the methodologies used in our investigation, analyze the results, and discuss the findings. We first summarize various fixed-point designs for the pipeline building component, the multiplier-adder pair. We then present the design considerations about the system integration. Various parameters are investigated in our research: pipelined stages, timing, silicon area. Additionally, the experiences and difficulties of using timing verifiers that are built in the silicon compiler will be discussed as well.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to acknowledge my parents and my wife for their unending support and patience. I would like to thank the staff of the Computer Lab in the Electrical and Computer Engineering Department at the Naval Postgraduate School and the manager of the Genesil Silicon Compiler technical support group.

I. INTRODUCTION

A. BACKGROUND

VLSI (Very Large Scale Integrated) circuit technology has resulted in a dramatic increase in the circuit density (number of components, gates, circuits or memory bits) contained within a single chip. This increase in the number of circuits has helped the development of application specific integrated circuit (ASIC) design.

The methods of ASIC design include full-custom design, gate-array design, standard cell circuit design, and silicon compilation methods. Silicon compilation is the newest method of ASIC design and allows the designer a higher degree of convenience than other methods. The silicon compiler works from a high-level description of the circuit that allows the designer to perform successive design iterations quickly and efficiently, providing the designer rapid access to key parameters such as chip size, power consumption, and timing constraints.

An algorithm based on the pipelined Infinite Impulse Response (IIR) filter has been suggested in the literature [Ref. 1]. That paper developed the technique and demonstrated the existence of a stable high-rate pipeline realization of a practically useful filter. Jangsri has presented the realization of the high speed IIR notch filter using the pipelined technique of Loomis and Sinha [Ref. 2]. Figure 1.1 shows the basic second-order IIR notch filter. From Figure 1.1 we

can see that the pipeline multiply-add unit is fundamental building block to the pipeline digital filter. Therefore, the purpose of this thesis is to investigate alternative designs for the pipeline multiply-add unit and to design the VLSI implementation of a pipelined notch filter using the Genesil silicon compiler.

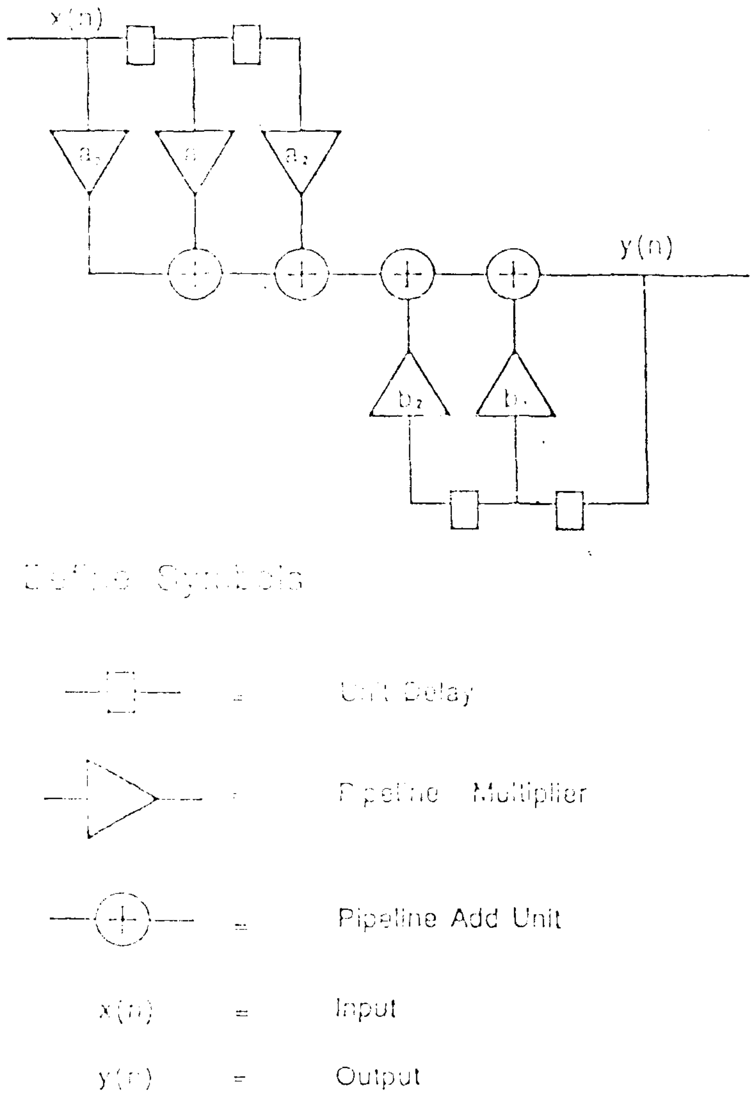


Figure 1.1 Second Order Notch Filter [from Ref. 2, P. 7]

B. NOTCH FILTER AND PIPELINED DESIGN

In a telecommunication system, the receiver signal contains both a desired component and undesired components as in Figure 1.2. The undesired or unwanted components are often called interference. In an attempt to recover the desired signal, we can put the signal received through a filter as in Figure 1.3.

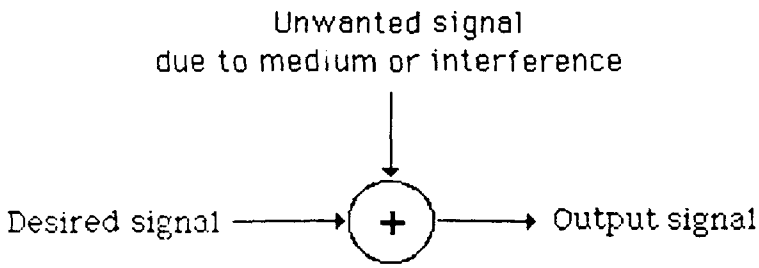


Figure 1.2 Generation of A Signal Plus Interference

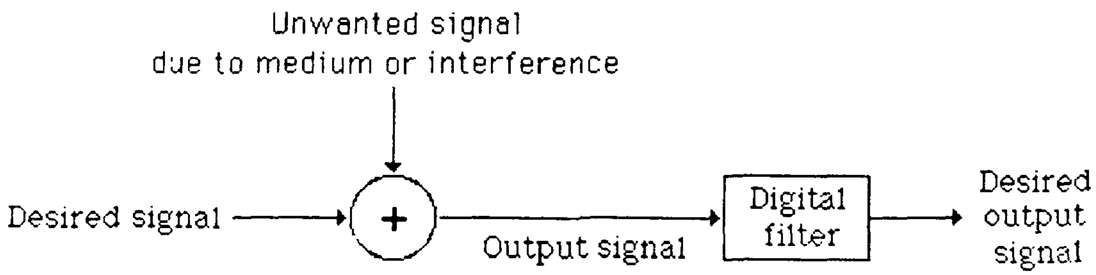


Figure 1.3 Noise System with Filter

No modern discussion of filter classifications would be complete without some mention of the concept of the digital filter. There are few areas of the electrical field that have not been affected by the

rapid evolution of modern computer technology, and filter technology is no exception. A digital filter is an implementation of a numerical algorithm that transforms a sampled input data signal into an output data signal, with a desired filtering objective accomplished through the data processing. Most digital filters employ linear constant coefficient difference equations that relate the output data stream to the input data stream.

A possible system employing a digital filter is illustrated in Figure 1.4. The input analog signal is applied to an analog-to-digital converter, which converters the analog signal into a sequence of digital numbers (or words). The signal is then processed by the digital filter. The output is then converted back to an analog signal by means of a digital-to-analog converter.

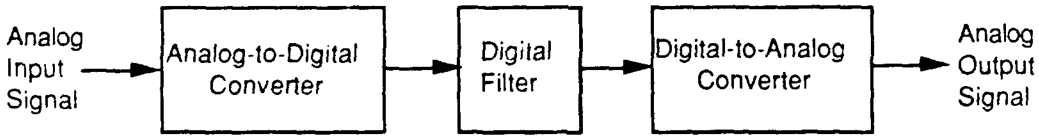


Figure 1.4 Digital Filter with Analog Input and Output

Sometimes, digital filters are used as a part of a larger digital processing system, one that will not necessarily produce an analog output. With some systems such as Figure 1.5 shown, the input might be in analog form initially, and/or the output might be desired in a digital form, in which case one of the conversions could be eliminated.

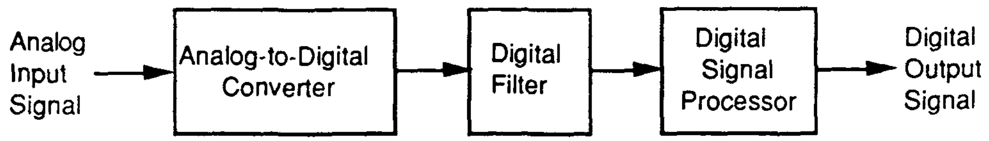


Figure 1.5 Digital Filter with Analog Input and Digital Output

A notch filter is a narrowband-rejection filter that produces a sharp notch in the frequency response curve of a system and found in many television transmitters and superheterodyne receivers. Generally it is inserted in one of the intermediate-frequency (IF) stages of a superheterodyne receiver, where the bandpass frequency is constant. The notch filter is extremely convenient to provide attenuation at the low-frequency end of the channel as well as for reducing interference caused by strong, unmodulated carriers within the bandpass of a receiver. Figure 1.6 shows the characteristic of a notch filter.

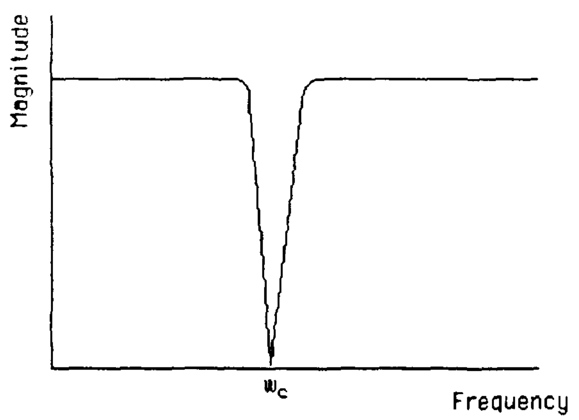


Figure 1.6 The Characteristic of Notch Filter

The starting point in the discussion of digital filtering of signals is the concept of the sampled-data signal. A sampled-data signal is one that consists of a regular sequence of encoded samples of a continuous-time or analog signal. As a result of Shannon's sampling theorem [Ref. 3], to be able to recover the original signal from the sampled-data signal, it is necessary that no portion of the spectrum of the first translated component overlap the original spectrum. It means that the sampling rate, which is also the clock rate of the filter, must be at least as high as twice the highest frequency in a spectrum. Thus the use of digital filters in real-time application involving high frequencies demands a high sample rate. In other words, the use of digital filters in real-time applications, such as spaceborne applications, involving high frequencies demands a high sampling rate.

Which techniques will yield the highest throughput or clock rate possible with a given state-of-the-art? Pipeline techniques are typically used to increase the clock rate of a particular digital system. In order to demonstrate the use of pipeline techniques we investigated the IIR designs in this research. Specifically we designed a pipelined multiply-add unit which is the heart of the IIR designs investigated.

C. GENESIL SILICON COMPILATION

Silicon processing technology has advanced so rapidly that hundreds of thousands of transistors can now be put on a chip. The advance continues apace. The problems of designing chips of current

complexity are already immense, so better design techniques are needed to use this emerging technology effectively. The Genesil silicon compiler system is a design automation system which provides the user with the capability of designing VLSI circuits from a high-level system description to manufacture tapeout by producing the IC circuits from architectural descriptions. The silicon compiler is effective because it contains all of the components necessary for circuit design within one tool. It allows the system designer to quickly and effectively create workable circuits and investigate architectural alternatives. Once the designer has specified his design, the silicon compiler synthesizes its layout. Additionally, simulation models, timing analysis models, and test generation models can be prepared. There are three previous theses that describe the use of Genesil [Refs. 4, 5 and 6] and they are highly recommended as background reading for anyone desiring to use the system.

D. THESIS GOALS AND ORGANIZATION

The main goal of this thesis is to design the VLSI implementation of the pipelined multiply-add component of the pipelined notch filter using the Genesil Silicon Compiler. Applying Loomis/Sinha pipelined methods to design a pipelined notch filter has been investigated previously at the Naval Postgraduate School by Jangsri [Refs. 1 and 2]. The investigation was thorough and the basic design was developed. In Chapter II, we investigate various systems for implementing the basic building block that is used by the Jangsri pipelined notched filter structure: the ones' complement system,

two's complement system, and signed magnitude system. Chapter II also includes timing and design verification studies. We will describe the implementation of pipelined notch filter in Chapter III. Chapter III also includes the implementation of the basic pipelined building block and chip construction. Chapter IV will present a summary of the work completed and the conclusions drawn from this research.

II. BASIC MODULE ARCHITECTURE

A. INTRODUCTION

A notch filter is sometimes called a band-rejection filter. It can remove certain preselected frequencies from the frequency-response output. It is useful in telecommunication systems to remove undesired interference frequencies. With the CAD tools such as silicon compilers and software design validation, we can explore and rapidly prototype our design alternatives before fabrication. Pipelined implementation of notch filters are examined in this research.

B. ASSUMPTIONS OF PIPELINE NOTCH FILTER DESIGN

The assumptions of pipelined notch filter design were composed of six main parts. They were:

1. **The input and output data type must be specified by giving a signed fixed-point binary number of 14 or more bits.**

The input and output (I/O) sequence represents the value of a sinusoidal signal that is sampled at given points in time. This can be accomplished by an analog to digital (A/D) converter and digital to analog (D/A) converter external to the chip.

There are two possible ways of specifying the I/O sequence: by giving it a fixed-point position or by employing a floating-point representation. In either case, the binary point is not physically

visible but is assumed from the fact that the number stored in the register is treated as a fraction or as an integer. If the I/O data was represented by floating-point as IEEE standard format, the I/O pad (pin) count will increase and cost will higher than for the fixed-point format. The 14 or more bits provides sufficient significant figures for proper data representation and filter operation.

2. The position ot the binary point is needed to represent mixed integer-fraction numbers.

From Jangsri [Ref. 2] the range of data Y[t] and Z[t] is between -4 and 4, the range of data a(i) and b(i) is between -2 to 2, and range of data X[t] is between -1 to 1. Therefore, the position of the binary point needs to be to the right of the sign and two most significant bits. As mentioned above, we denote the fixed-point format of X[t], a(i) ,and Z[t] as $X_s.X_1X_2.....X_{13}$, $a_sa_0.a_1a_2.....a_{12}$, and $Z_sZ_{-1}Z_0.Z_1Z_2.....Z_{13}$.

3. The magnitude of negative number X[t], Y[t] and Z[t] is represented by X's complement.

The input represents the value of a sinusoidal signal that is sampled at given points in time. This can be accomplished by an analog to digital (A/D) converter external to the chip. It was also assumed that this input was in the form of a ones' or two's complement number.

4. **The coefficients a(i) and b(i) are user adjusted parameters, and represented in signed magnitude form.**

These coefficients determine the rejected frequency and notch width.

5. **All flip-flop control and clock signals are external and their clock period equals maximum stage timing delay.**

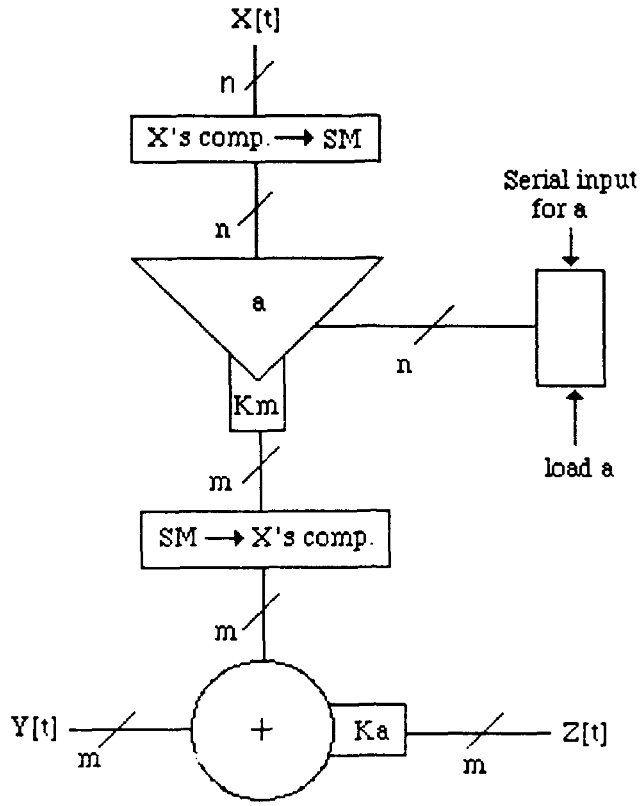
The timing for the entire system is controlled by the master-clock generator whose clock pulses are applied to all flip-flops in the system. Therefore we assume the master-clock generator is an external unit and its clock pulse period equals maximum stage timing delay.

6. **All stages in the pipeline receive the same clock pulse simultaneously.**

Due to circuit lengths, loading, and driver circuits, it is nearly impossible to guarantee that all stages of a pipelined circuit receive the same pulse at exactly the same time. Nevertheless we assume this situation is not existent in this thesis.

C. MULTIPLY-ADD MODULE

The multiply-add module is the basic building block that is used by the pipelined notch filter. We now discuss alternative designs of this building block and will elaborate its pipelined versions in the next section. Figure 2.1 shows the basic multiply-add module.



Definitions

$a = a_s a_0 . a_1 \ldots\ldots a_{n-2}$

$X[t] = x_s . x_1 x_2 \ldots\ldots x_{n-1}$

$Z[t] = z_s z_{-1} z_0 . z_1 z_2 \ldots\ldots z_{n-3}$

Figure 2.1 Multiply-Add Module

The triangle indicates the multiplier and the circle represents the adder. The multiply-add module takes fixed-point inputs (X[t]), multiplies by a constant (A), adds with another input stream (Y[t]), and then produces the output stream (Z[t]). That is, Z[t] = a * X[t] + Y[t]. The meaning of Km and Ka are number of pipelined stages for multiplier section (Km) and adder section (Ka) respectively. We will

discuss Km and Ka in Chapter III. There are four number systems for representing negative numbers: signed magnitude, ones' complement, two's complement, and excess 2^{m-1}. The range of excess representation is exactly the same as that of two's complement numbers. In fact, the representation of any number in the two systems are identical except for the sign bits, which are always opposite [Ref. 7]. Traditionally the excess representation is used in the exponent of floating-point number systems. Therefore, we investigated all representation except the excess system.

An add-multiply module consists of six blocks. They are: coefficient block, complement to signed magnitude block, multiplier, signed magnitude to complement block, adder, and overflow block. Note that if the input stream is in signed magnitude format then we do not need the complement to signed magnitude block.

1. Coefficient Block

To reduce the pin requirement we decided to load the coefficient serially so that we need only two pins (one for information, one for control) for loading one coefficient. Therefore, we design this block as a serial-in-parallel-out register. Figure 2.2 shows a four-bit serial-in-parallel-out schematic for the coefficient block. In the notch filter design we need a 14-bit coefficient constant (a). The constant loaded into this block could be either position or negative. We denote the fixed-point format of a constant a as $a=a_s a_0 . a_1 a_2 \ldots . a_{12}$, where a_s is the sign bit.

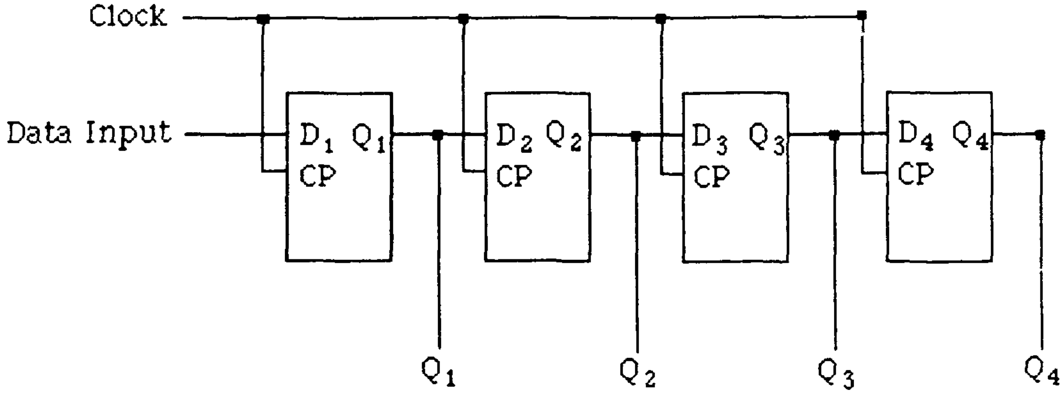


Figure 2.2 Serial In Parallel Out

2. Complement to Signed Magnitude Block

In the three number systems considered, the sign bit is identical for each system and positioned as the leftmost bit. For positive numbers the three number systems are identical. For negative numbers, *in ones' complement* representation the magnitude of a number is represented by that number's ones' complement. Therefore, the conversion from ones' complement to signed magnitude is quite straightforward: for position numbers do nothing, for negative number take the ones' complement. We can achieve this function simply by examining the sign bit and selectively complementing the magnitude through exclusive-or (XOR) gates. A two-input XOR gate has no effect when one of the inputs is zero and acts as an inverter when one of the inputs is one. Figure 2.3 shows the logic design for converting ones' complement to signed magnitude format.

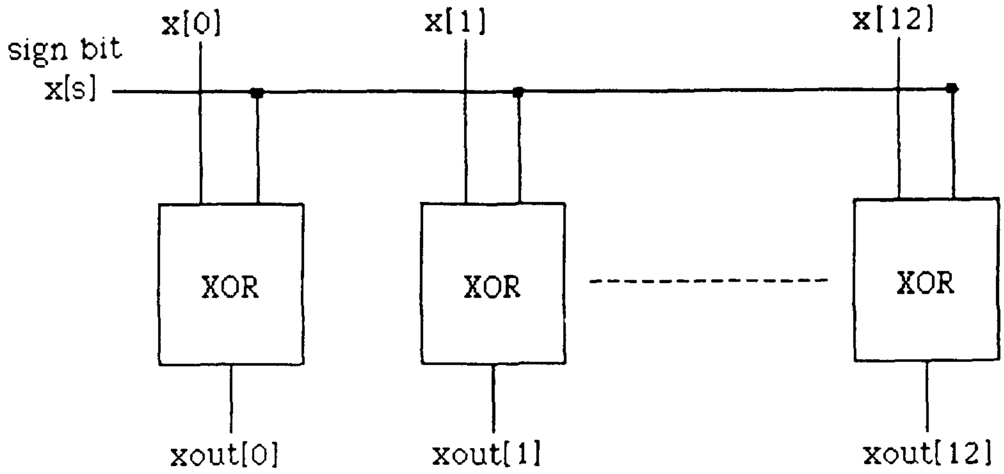


Figure 2.3 Ones' Complement Convert to Sign-Magnitude Block

For a negative number, by definition, if we add one to the ones' complement representation we get the two's complement representation. Given a negative number in two's complement form if we take the ones' complement of it, excluding the sign bit, the resulting magnitude is one less than the correct magnitude. Therefore, for a negative number the conversion from two's complement to signed-magnitude is simply a two-step operation: (1) take ones' complement of the magnitude value (2) add one to it. Again, for a two's complement number we can achieve this by XOR the sign bit with the magnitude bits. Since for a positive number, i.e., when the sign bit is 0, the XOR of sign bit with magnitude bits takes no effect. For a negative number, i.e., when the sign bit is 1, the XOR of sign bit with magnitude bits is equivalent to the effect of taking ones' complement of them. The operation of adding one (coming

from the sign bit) is to fulfill the two's complement requirement. For example, if we have a 4-bit representation for -3 the two's complement gives 1101 and sign-magnitude gives 1011. We get the number 1010 at step (1) and the number 1011 at step (2). The logic design of this part is an array of XOR gates and one adder. Note that when the sign bit is zero the operation of adding the sign bit to the magnitude converting to two's complement has no effect.

3. Multiplier Block

The Genesil library multiplier block array is an array of half and full adders which provides a parallel multiplier typically used for unsigned integer multiplication [Ref. 8] and requires external circuitry for signed operations. The least significant bits (LSB) are produced directly from the array, but an external adder is required to complete the partial product addition of the most significant bits (MSB). The multiplier and multiplicand widths can be varied from 4 to 32 bits, but the multiplier width cannot exceed the multiplicand width. At the same time, an external XOR is also required for sign bit operation.

4. Signed Magnitude to Complement Block

The logic design of this block is similar to that of the conversion from complement numbers to signed-magnitude. For positive numbers we do not have to do the conversion. For negative numbers in ones' complement format, we simply examine the sign bit and take ones' complement of the magnitude value. If the negative number is in two's complement format, we need an

additional step of adding one (the sign bit) to its ones' complement (excluding sign bit).

5. Full Adder Block

A full adder was extracted from the Genesil Compiler Library, Volume III [Ref. 9], which is a collection of all system random logic blocks available. The Genesil Silicon Compiler system library full adder can be programmed, via a menu, from 1 to 16 bits in width. It has two data input buses and a single carry input which are added together resulting in the data output bus and a carry output. As mentioned above, the 16-bit width is selected on the module.

7. Overflow Block

The purpose of this block was to detect overflow condition of each module on the chip of notch filter. The overflow condition can occur only when both numbers are positive or both numbers are negative. Thus, overflow can be said to have occured if the sign of the resultant differs from that of the original numbers.

D. TIMING ANALYSIS AND PIPELINED DESIGN

In general, we can design a pipelined notch filter by using pipelined multiply-add modules. A pipelined multiply-add module consists of a Km-stage pipelined multiply and Ka-stage pipelined adder. The pipelined multiply-add module was designed with all three number systems in CMOS 1-micron technology. Figures 2.4, 2.5 and 2.6 show multiply-add modules with all three number systems. Their logic validation was performed by using the Functional

Simulation, and Timing analysis was performed to investigate the worst case delay in each module.

Table 2.1 shows the timing analysis and silicon area required for unpipelined version, Km=0 and Ka=0. From Table 2.1 we note that the signed magnitude format does not provide any favorable consideration both in speed and silicon area. We therefore dropped the signed-magnitude in further discussion. One interesting fact to note in Table 1 is that the two's complement format requires larger silicon area than ones' complement. This is due to the fact that every time when we convert a negative number into two's complement form we need to convert it into ones' complement and then add one to it. For two's complement, the "Add one" operation implies that we need one more adder than ones' complement. The timing of two's complement format is slightly slower than the ones' complement.

TABLE 2.1: Size and Timing for Unpipelined Multiply-Add Module

	Total Area (square mils)	Maximum Propagation Delay (ns)
signed magnitude	23220.34	66.8
ones' complement	11309.56	62.5
two's complement	16349.52	68.4

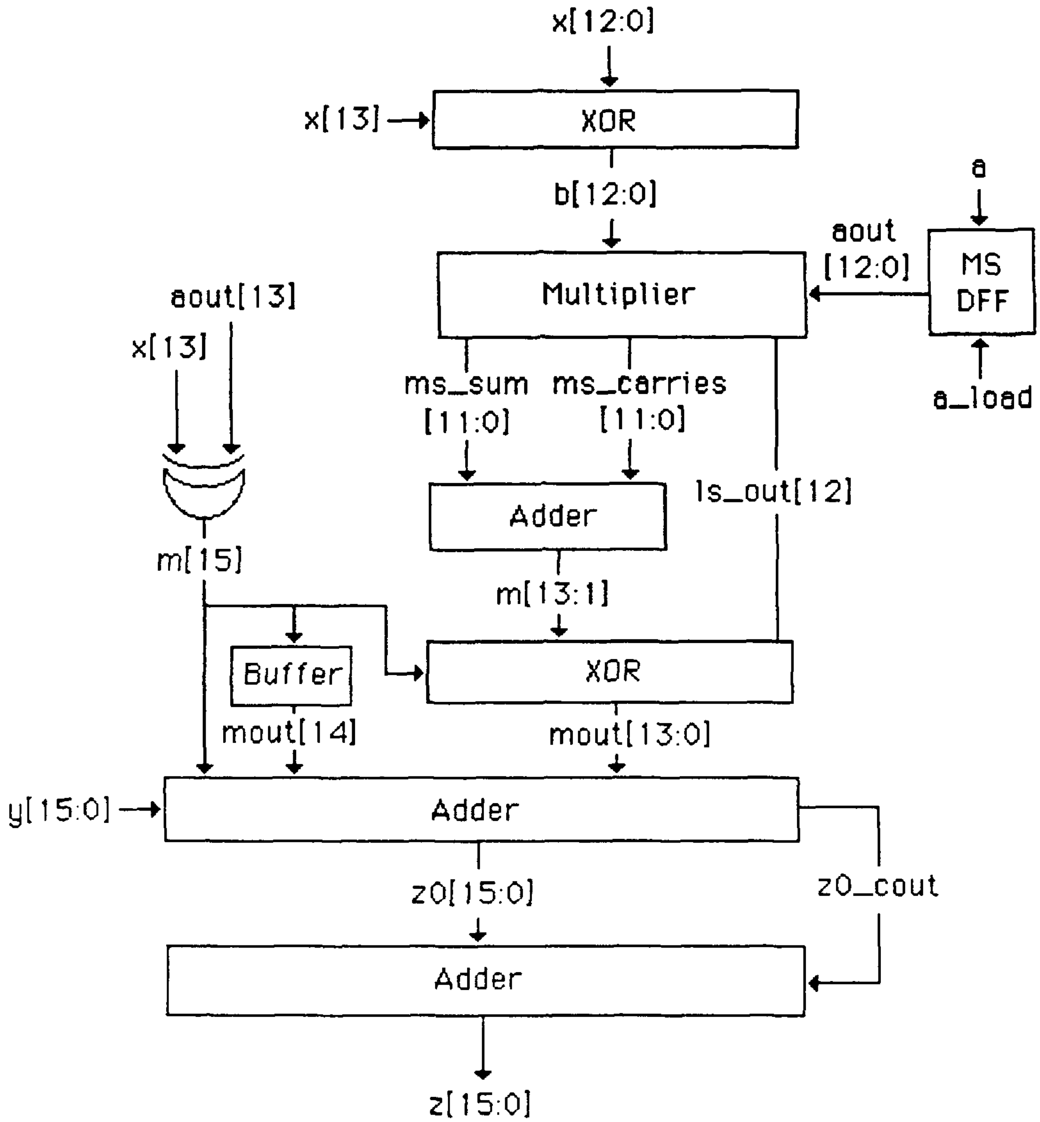


Figure 2.4 Ones' Complement Unpipelined Multiply-Add Module

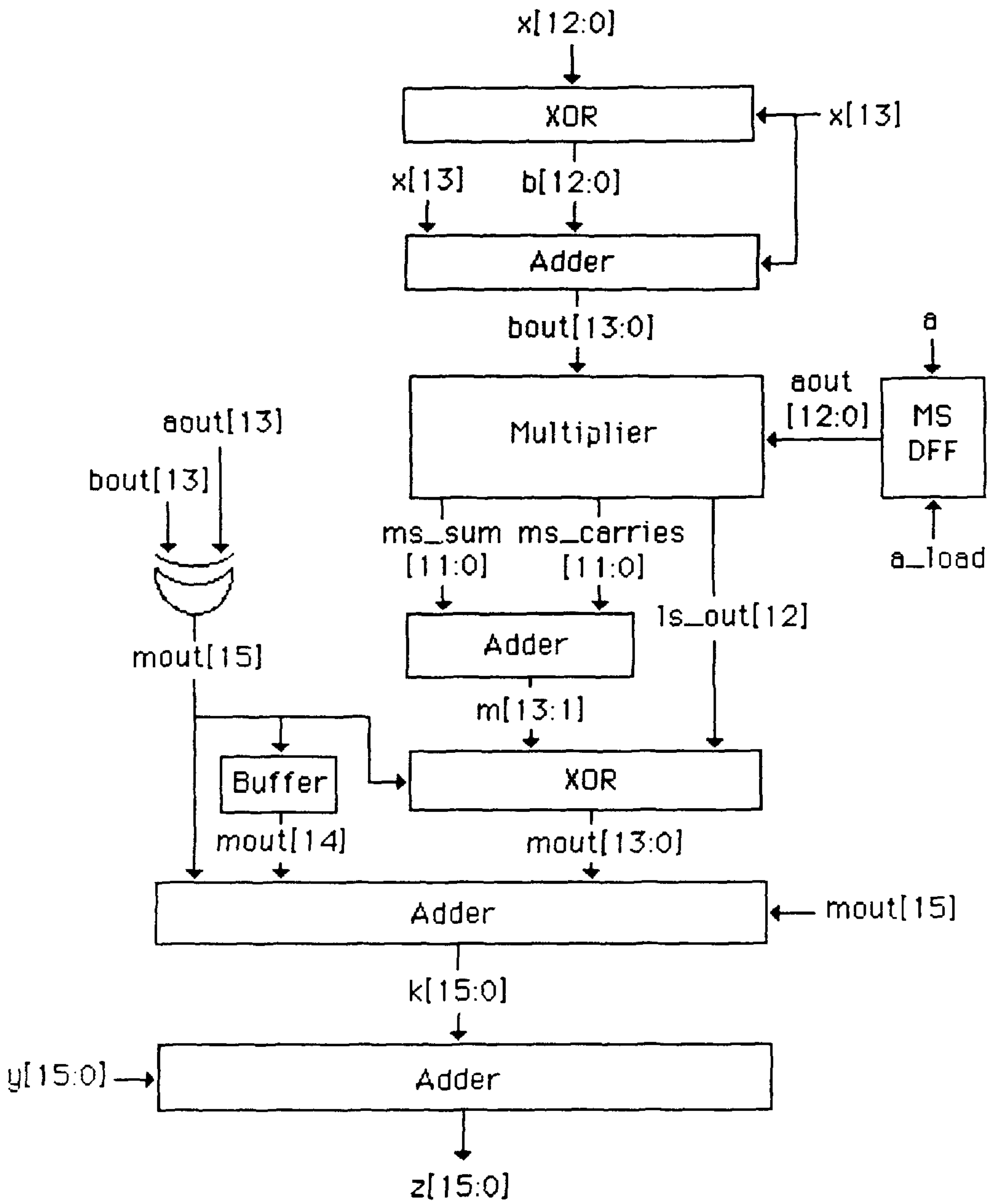


Figure 2.5 Two's Complement Unpipelined Multiply-Add Module

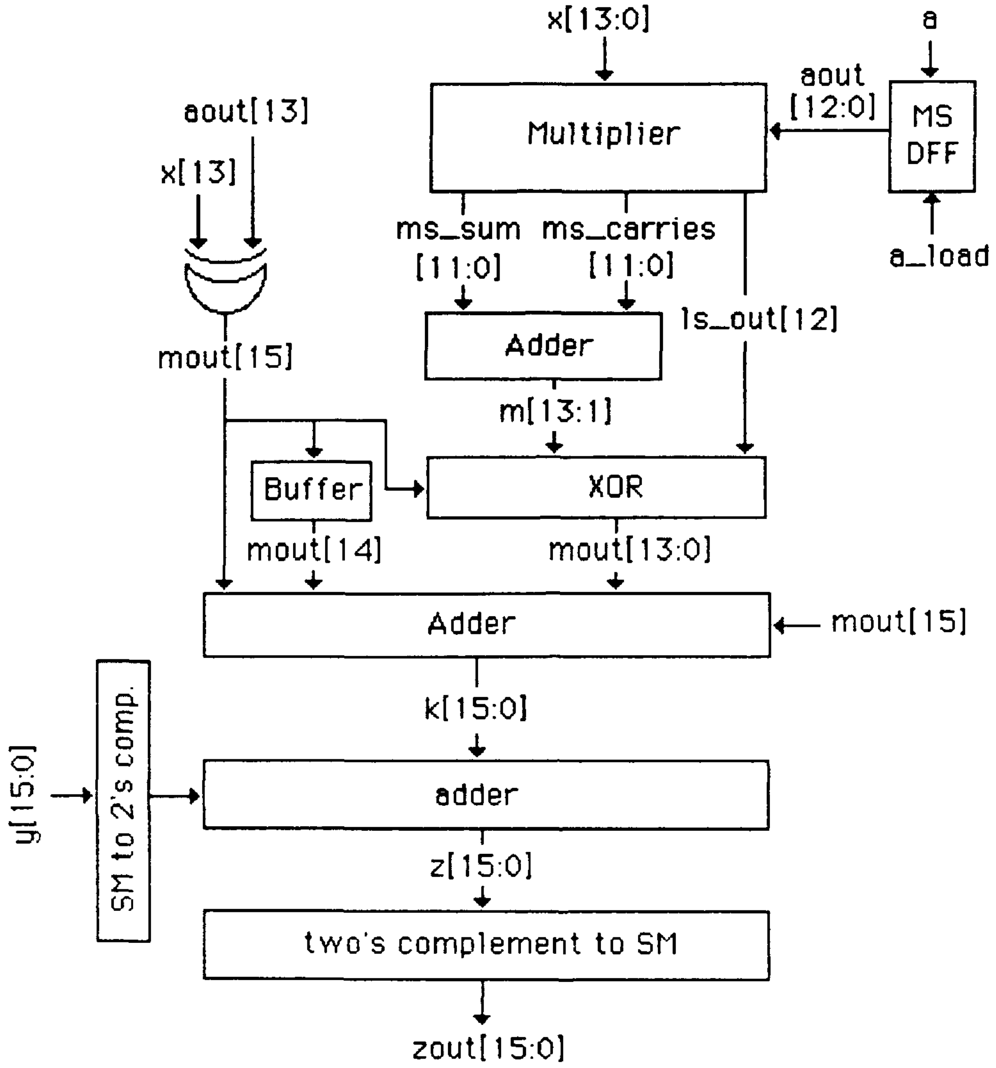


Figure 2.5 Signed Magnitude Unpipelined Multiply-Add Module

III. IMPLEMENTATION OF PIPELINED NOTCH FILTER

A. PIPELINED CIRCUIT FOR HIGH PERFORMANCE

Pipelining generally takes the approach of splitting the function to be performed into sub-functions or smaller pieces and allocating separate hardware to each piece. Each piece or subfunction is defined as a stage. The purpose of pipelined circuits is to increase the throughput. Because (1) the use of digital filters in the real-time application involving high frequencies demands a high sampling rate and (2) the pipeline techniques yields the highest throughput or clock rate possible for a given state of the art [Ref 10 and 11], therefore we consider the application of pipeline techniques.

B. PIPELINED MULTIPLY-ADD MODULE

The pipeline design techniques splits a basic function into several subfunctions with five properties [Ref. 12]:

(1) Evaluation of the basic function is equivalent to some sequential evaluation of the subfunctions.

(2) The inputs for one subfunction come totally from outputs of previous subfunctions in the evaluation sequence.

(3) Other than the exchange of inputs and outputs, there are no interrelationships between subfunctions.

(4) Hardware may be developed to execute each subfunction.

(5) The times required for these hardware units to perform their individual evaluations usually approximately equal.

The following pipelined multiply-add modules were designed in CMOS 1-micron technology and with master-slave D flip-flops inserted for pipelining. In addition, the design of those modules was based on properties described above. The various pipelined designs of the multiply-add module we described in this chapter.

1. Two-stage pipelined multiply-add module

Figure 3.1 and 3.2 show the ones' complement and the two's complement pipelined multiply-add modules which have two-stage pipelined delay (Km=1, Ka=1). The Km means the number of stage pipelined delay on multiplier and Ka means the number of stages pipelined delay on adder. From the pipelined properties described above we may have alternative designs of splitting the module into stages. That is, we can insert the flip-flops at various points. In the ones' complement pipelined multiply-add module the first master-slave D flip-flop was inserted at the output of the external adder circuit of multiplier and the second flip-flop was inserted at the output of the adder block. In the two's complement pipelined multiply-add module the first flip-flop was inserted between the multiplier and the external adder circuit of multiplier and the second flip-flop position was at the end of adder block same as the ones' complement pipelined module.

Table 3.1 summarize the size and performance for two-stage pipelined version. It indicates that the slowest speed of stages in

ones' complement module is 43.7 ns and in two's complement module is 40.8 ns. Therefore the operating clock rate can be 22.9 MHz for ones' complement module and 24.5 MHz for two's complement module. The ones' complement module requires less silicon area smaller than the two's complement module.

TABLE 3.1 Size and Timing for Two-Stage Pipelined Multiply-Add Module

	Total Area (square mils)	Maximum Propagation Delay (ns)	
		stage 1	stage 2
ones' complement	25808.94	43.7	35.4
two's complement	27808.34	31.5	40.8

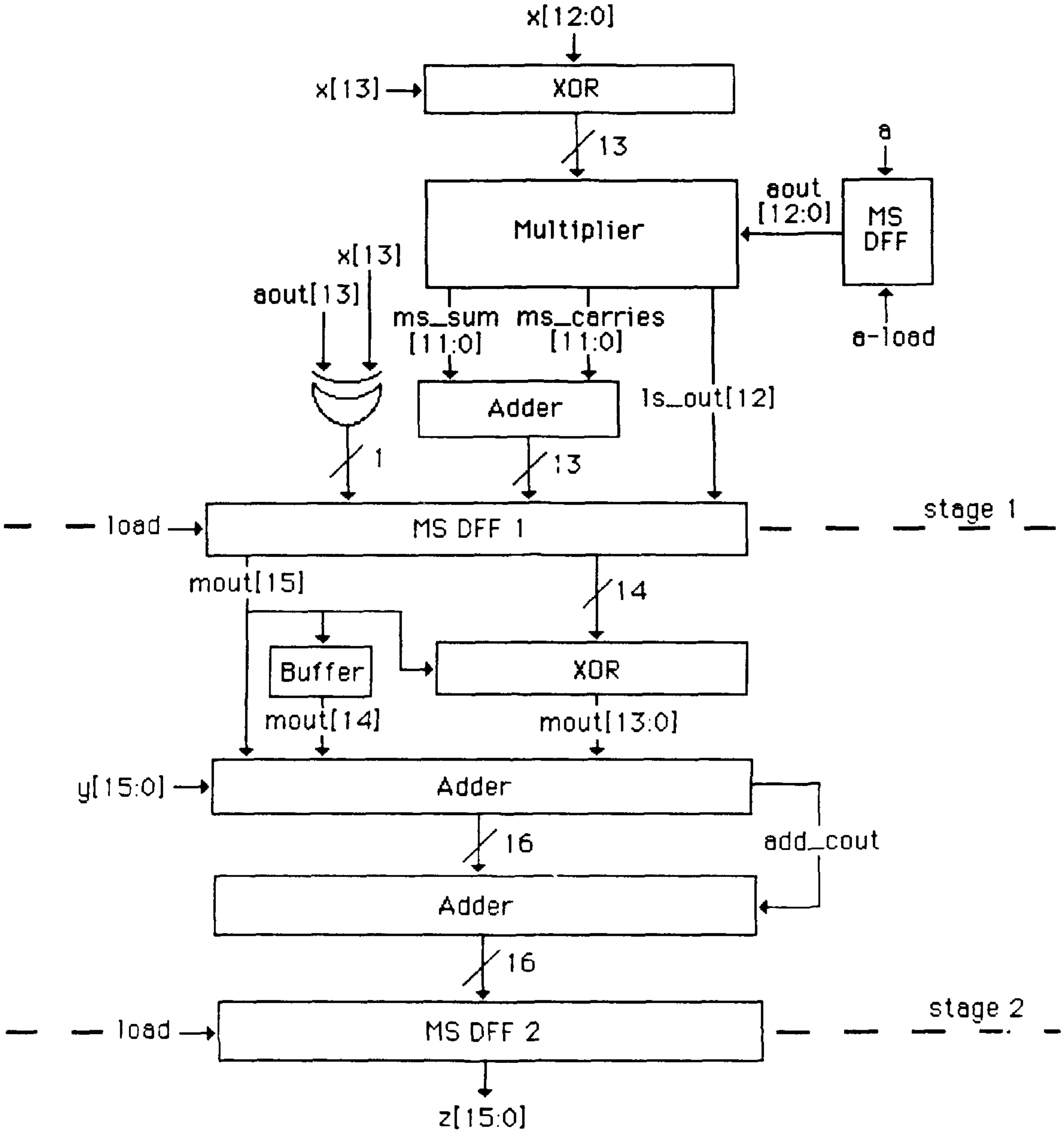


Figure 3.1 Ones' Complement Two-Stage Pipelined Multiply-Add Module

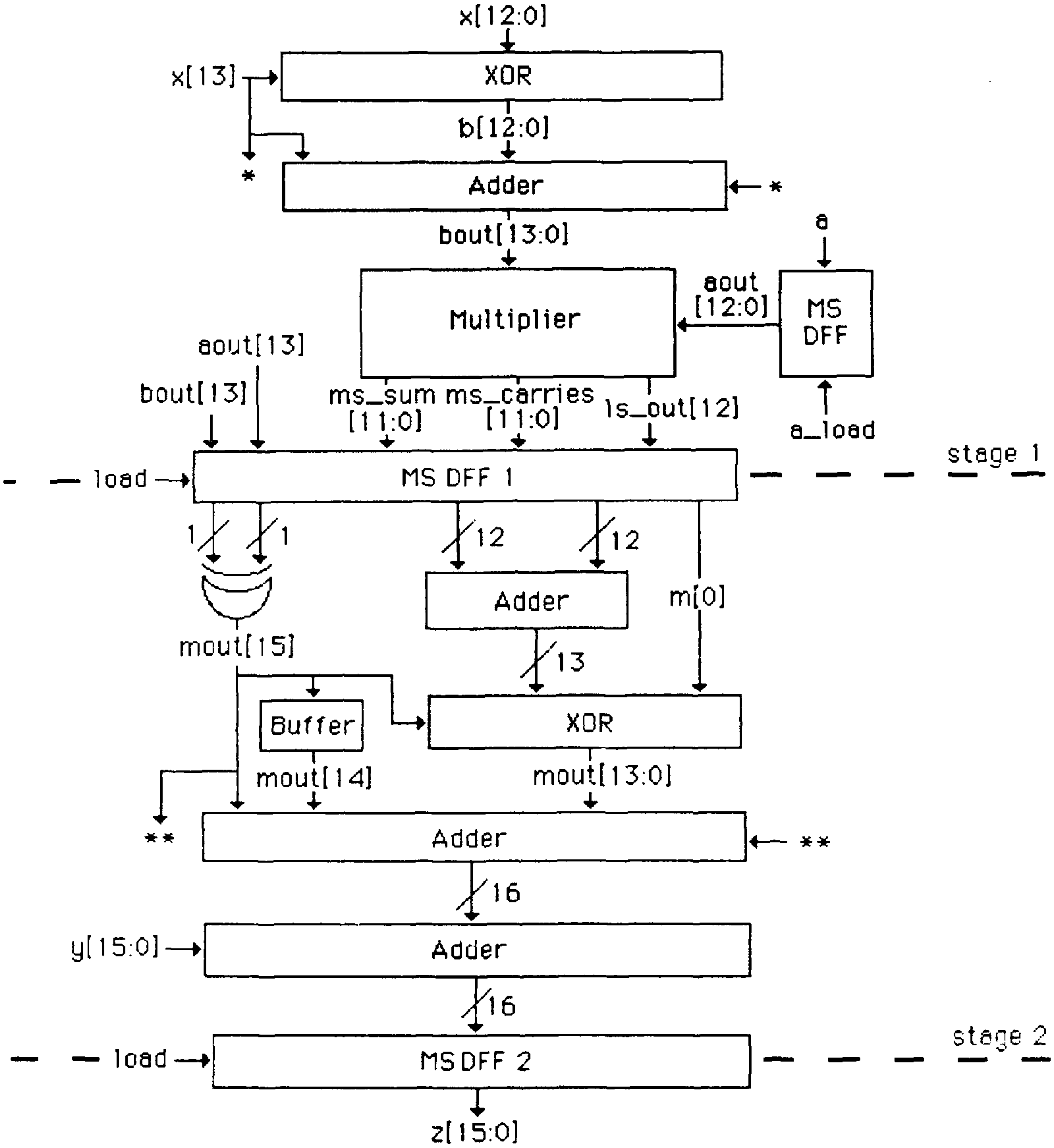


Figure 3.2 Two's Complement Two-Stage Pipeline Multiply-Add Module

2. Three-stage pipelined multiply-add module

Figure 3.3 and 3.4 show the ones' complement and the two's complement pipelined multiply-add module implemented as three-stage pipelined (Km=2, Ka=1). In these three-stage pipelined multiply-add modules, the three flip-flops are located at the same places in both the ones' complement and the two's complement pipelined module. The first flip-flop was inserted at the output of the multiplier, the second flip-flop was inserted at the output of the signed magnitude to complement block, and the third flip-flop was inserted at the output of the adder block.

Table 3.2 summarize the size and performance for these three-stage pipelined modules. The limiting speeds are 34.4 ns for ones' complement module and 34.3 ns for the two's complement module. In other words, we can achieve 28.6 MHz and 29.2 MHz clock speed for ones' complement and two's complement pipelined modules respectively. The ones' complement requires less silicon area than the two's complement module.

TABLE 3.2 Size and Timing for Three-Stage Pipelined Multiply-Add Module

	Total Area (square mils)	Maximum Propagation Delay (ns)		
		stage 1	stage 2	stage 3
Ones' complement	27892.0	25.1	20.6	34.9
Two's complement	36642.62	31.5	34.3	18.7

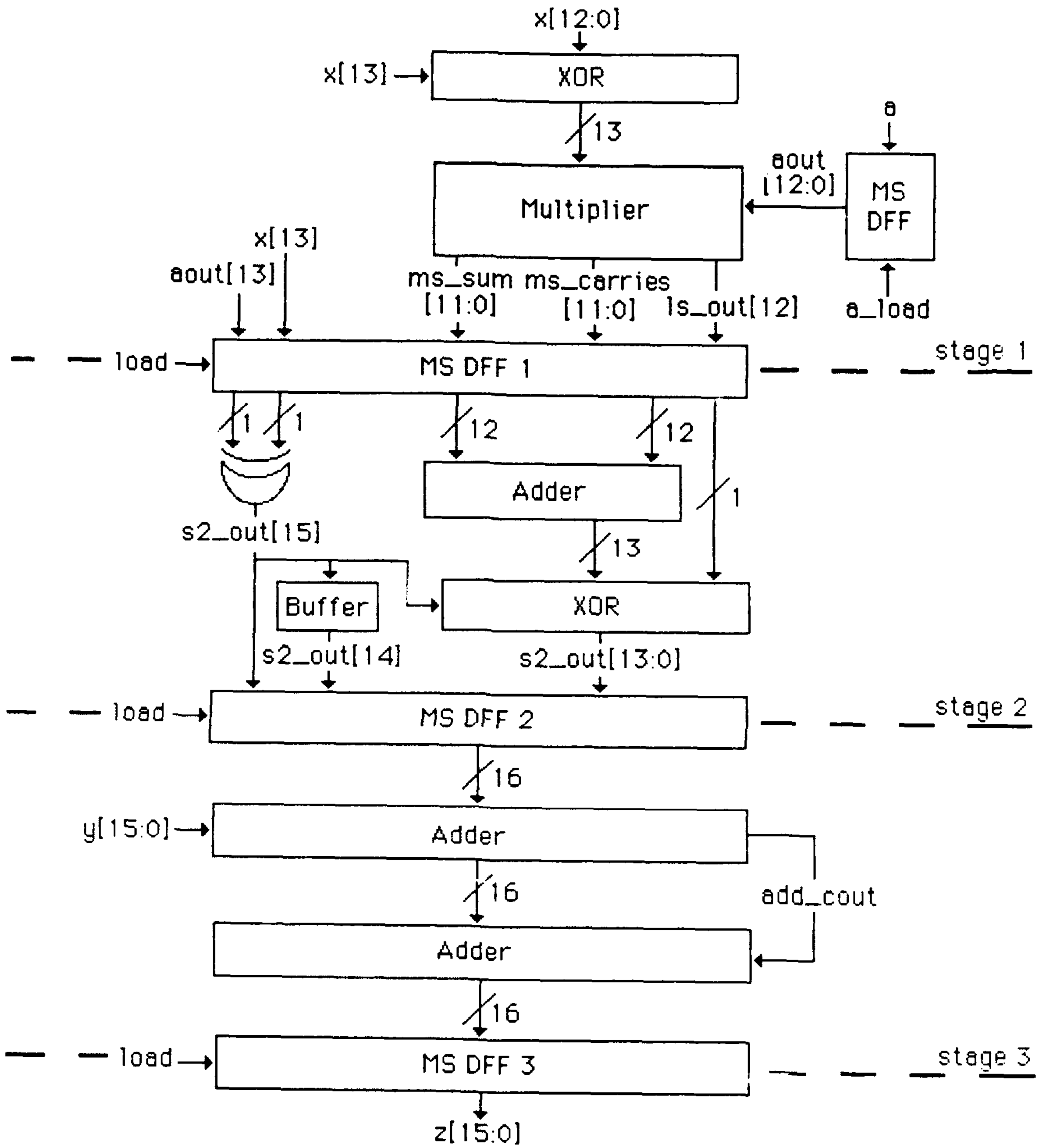


Figure 3.3 Ones' Complement Three-Stage Pipelined Multiply-Add Module

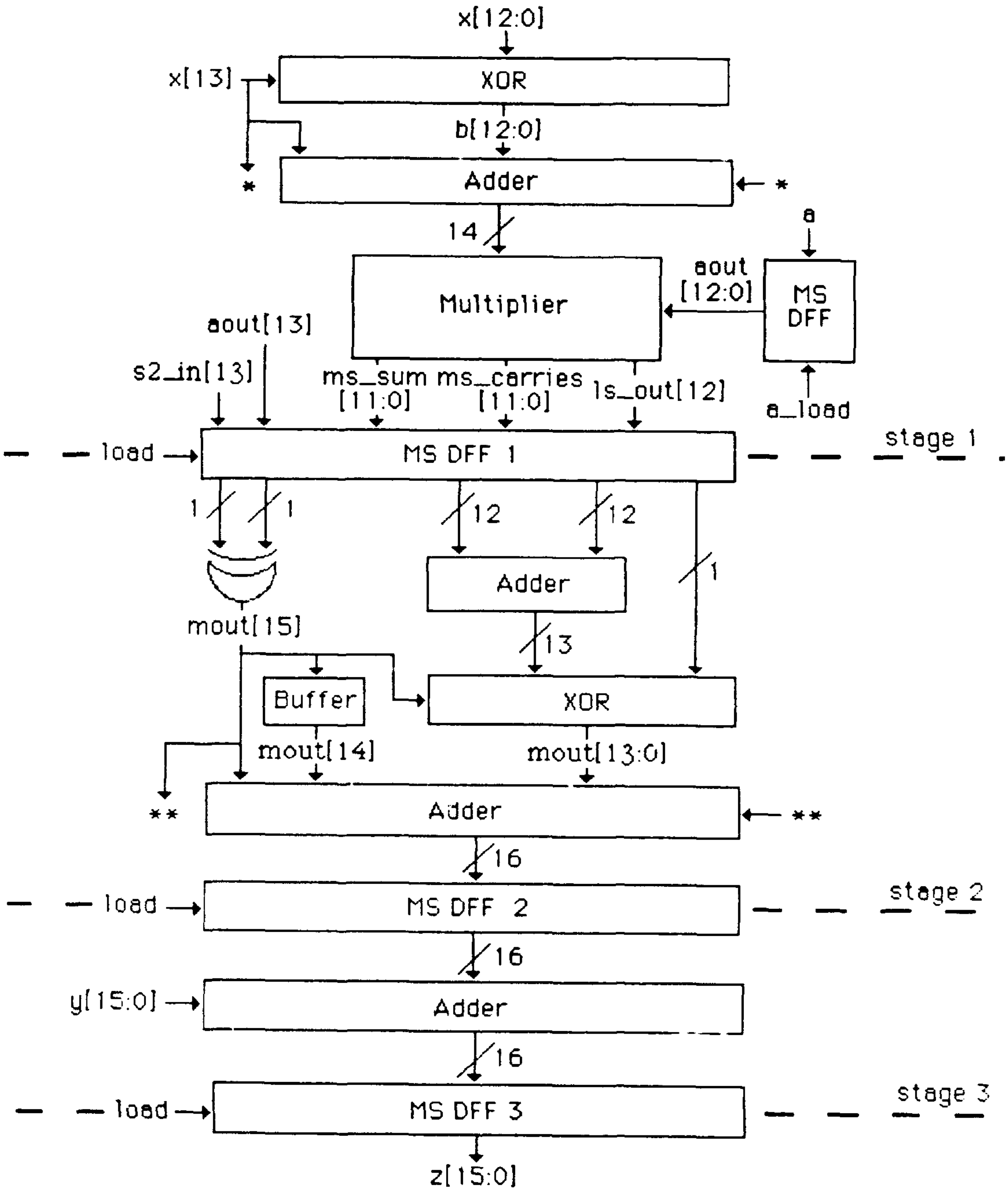


Figure 3.4 Two's Complement Three-Stage Pipelined Multiply-Add Module

3. Four-stage pipelined multiply-add module

Figure 3.5 shows the ones' complement four-stage pipelined multiply-add module (Km=2 and Ka=2). Compared with Figure 3.3, this module has inserted one more flip-flop into the adder block of the ones' complement three-stage pipelined module. Figure 3.6 shows the two's complement four-stage pipelined multiply-add module (Km=3 and Ka=1). Compared with Figure 3.4, this module has inserted one more flip-flop into the sign-magnitude to complement block of three-stage two's complement module.

Table 3.3 summarize the size and performance for these four-stage pipelined designs. It indicates that the lowest clock rate of stages is 25.1 ns for ones' complement module and 31.5 ns for the two's complement module. That is, we can achieve 39.8 MHz and 31.7 MHz clock speed these modules. The silicon area required by the ones' complement is smaller than two's complement.

In Chapter IV we will summarize and compare various pipelined modules' silicon area requirements and their clock rates.

TABLE 3.3 Size and Timing for Four-Stage Pipelined Multiply-Add Module

	Total Area (square mils)	Maximum Propagation Delay (ns)			
		stage 1	stage 2	stage 3	stage 4
Ones' complement	22072.95	25.1	20.6	19.4	19.4
Two's complement	50528.40	31.5	20.6	19.4	19.4

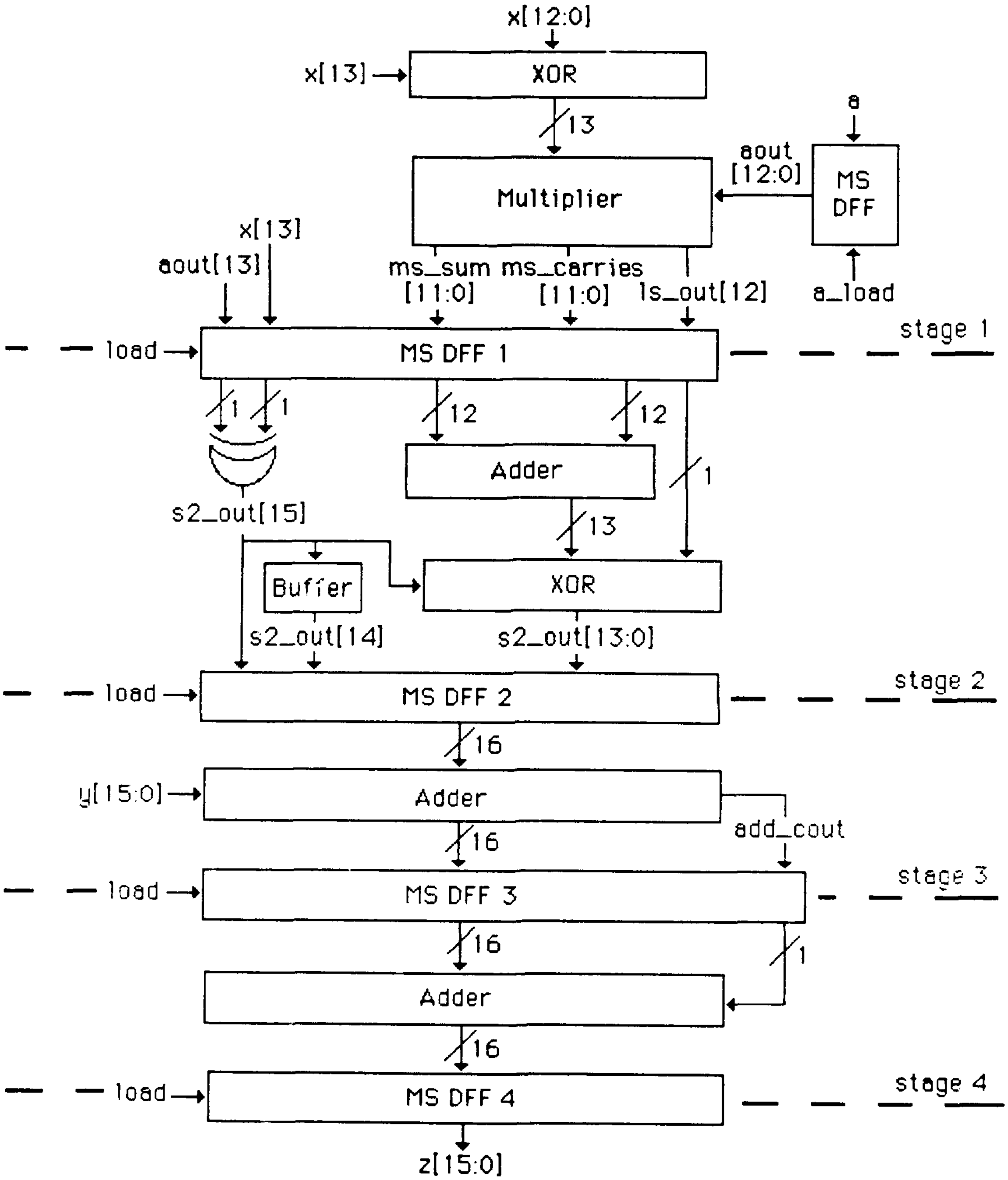


Figure 3.5 Ones' Complement Four-Stage Pipelined Multiply-Add Module

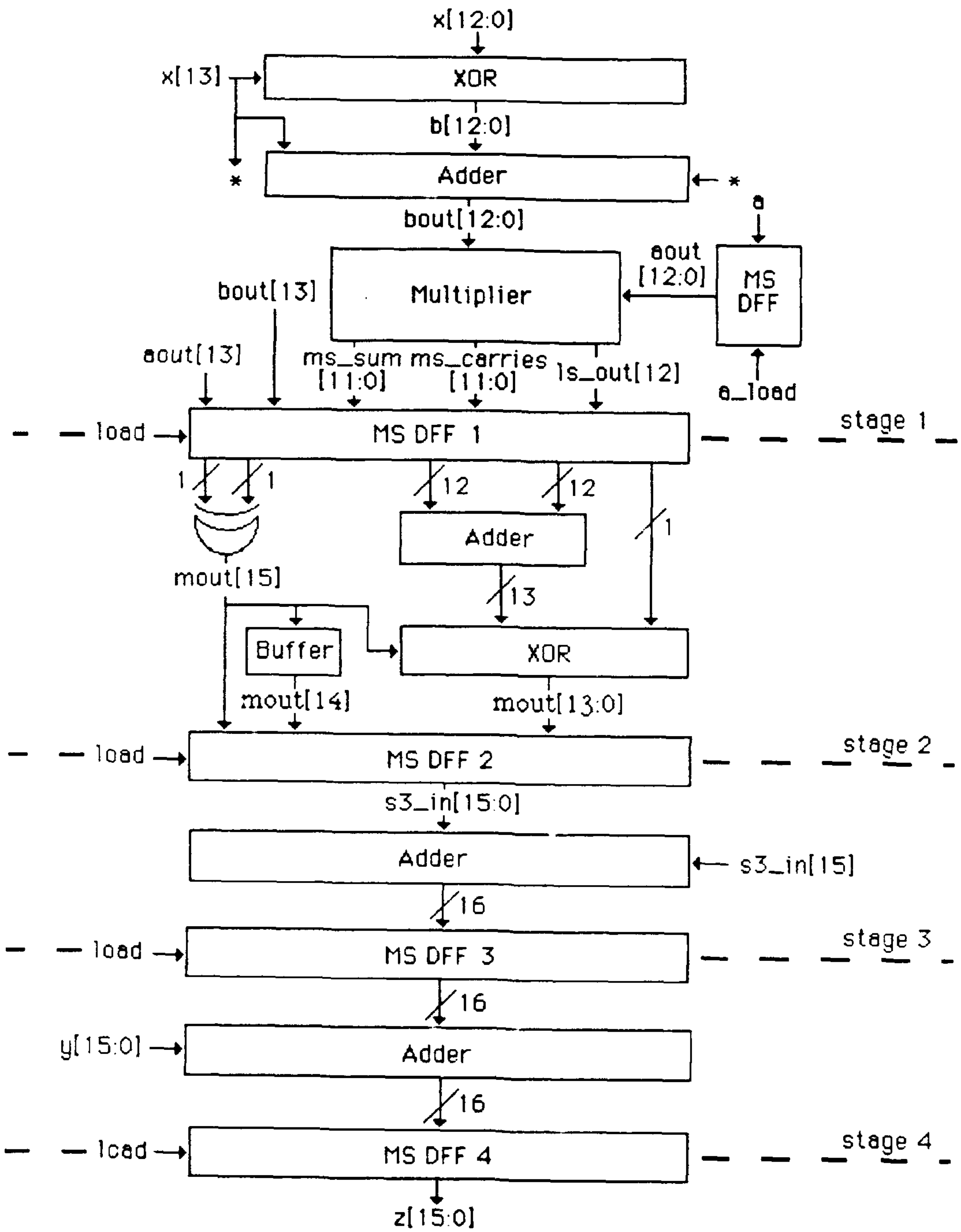


Figure 3.6 Two's Complement Four-Stage Pipelined Multiply-Add Module

4. Five-stage pipelined multiply-add module

Although the best strategy in pipeline design is to have the timing delay required approximately equal for all stages and no interrelationship between stages, we do not have much flexibility of splitting a component if we use the multiplier and adder that are generated by the Genesil. Figure 3.7 and 3.8 show the best partition of pipeline stages for the ones' and the two's complement pipelined modules respectively. The ones' complement module has three stages pipelined delay at multiplier (Km=3) and two stages pipelined delay at adder (Ka=2). And two's complement module has four stages pipelined delay at multiplier (Km=4) and one stage pipelined delay at adder (Ka=1). Table 3.4 summarize the performance results for five stages pipelined designs.

TABLE 3.4 Size and Timing for Five-Stage Pipelined Multiply-Add Module

	Total Area (square mils)	Maximum Propagation Delay (ns)				
		stage 1	stage 2	stage 3	stage 4	stage 5
Ones' complement	27383.0	1.5	21.4	20.6	19.4	19.4
Two's complement	33313.5	16.5	21.4	20.6	19.4	19.4

We have tried several other pipelined implementations and the fastest operational clock rate is limited by the multiplier stage.

This is because we use the parallel multiplier generated by the silicon compiler and we must use it as one individual block. The multiplier has 21 ns delay. This 21 ns delay limits the fastest clock rate of our notch filter approximately at 50 MHz (47.6 MHz) if we use the parallel multiplier that generated by the silicon compiler. Therefore, there is no point investigating multiply-add units of more than five stages.

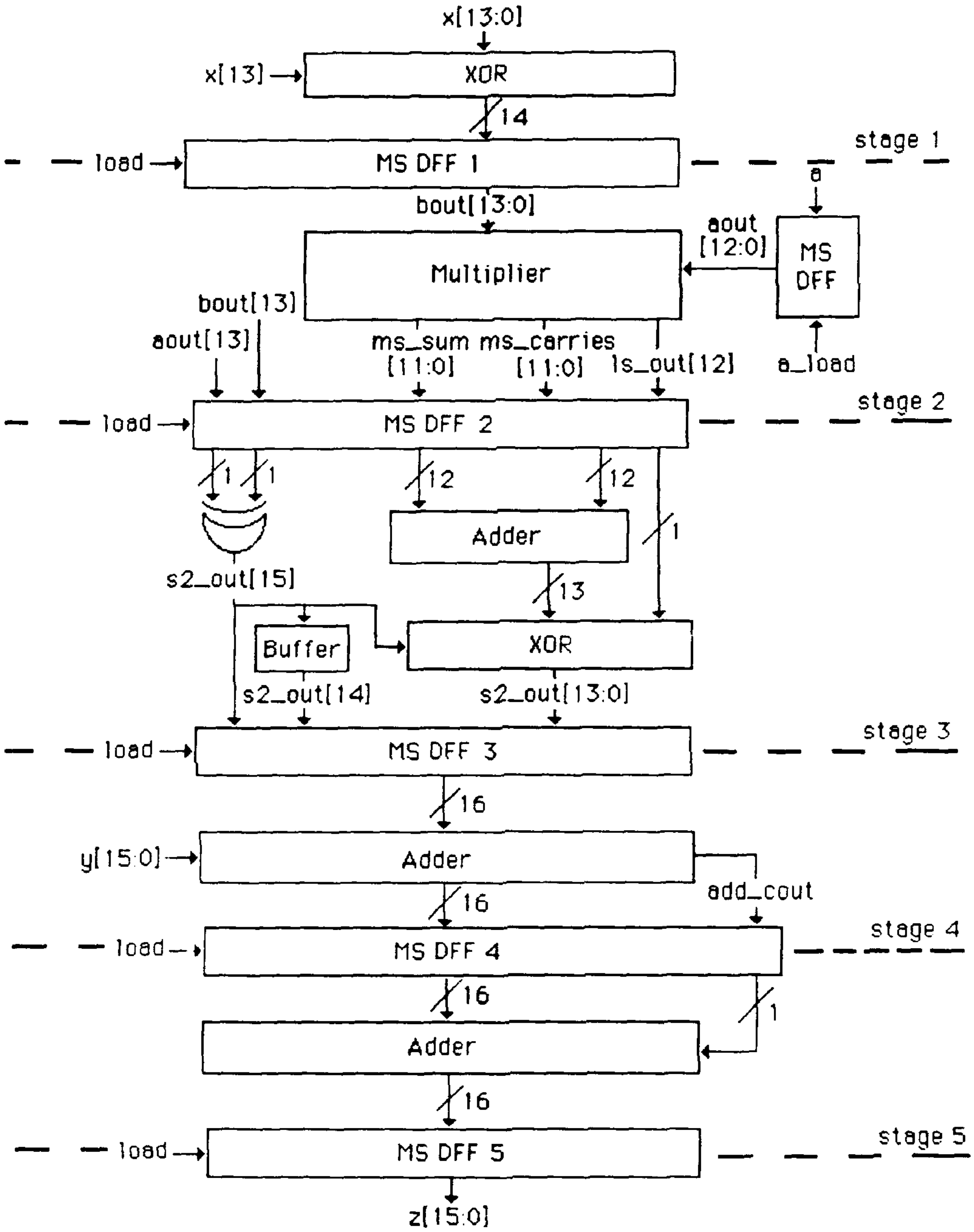


Figure 3.7 Ones' Complement Five-Stage Pipelined Multiply-Add Module

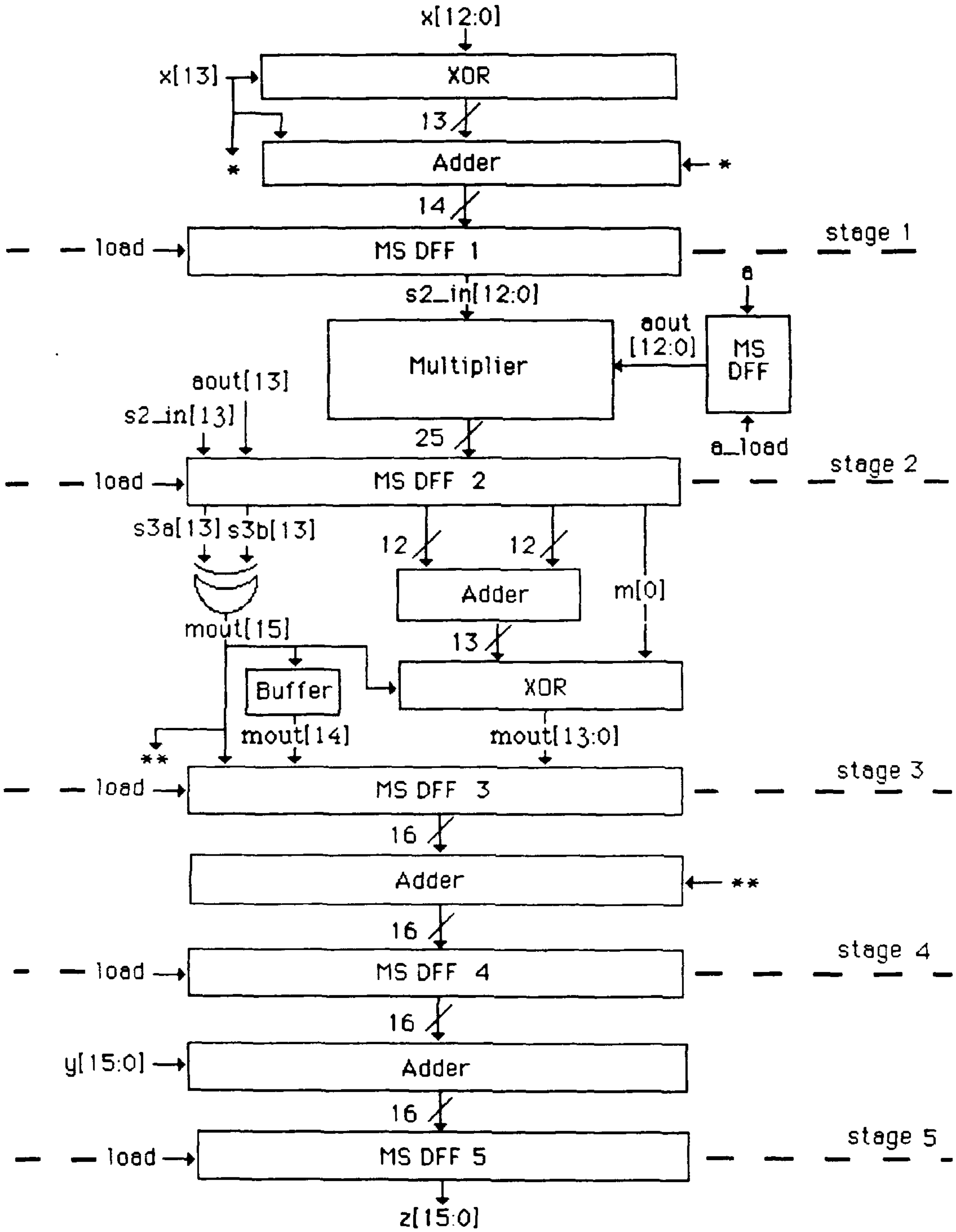


Figure 3.8 Two's Complement Five-Stage Pipelined Multiply-Add Module

IV. CONCLUSION

A. SUMMARY OF VARIOUS STAGE STRUCTURES

Table 4.1 summarizes of the size and timing of various designs of pipelined multiply-add modules. We can see the performance advantage in increased clock frequency gained by increasing the number of pipelined stages. We note, however, the performance enhancement comes with increased total chip area.

TABLE 4.1 Summary Timing and Size for Pipelined Modules

Number of Stages	Ones' complement		Two's complement	
	Total Area (Sq Mils)	Frequency (MHz)	Total Area (Sq Mils)	Frequency (MHz)
Unpipeline	11309.56	16.0	16349.52	14.6
2 Stages	25808.94	22.9	27808.34	24.5
3 Stages	27892.00	34.9	36642.62	34.3
4 Stages	22072.95	39.8	50528.40	31.7
5 Stages	27383.00	47.6	33313.50	47.6

The clock rate of the ones' complement pipelined module is increased from 16.0 MHz to 47.6 MHz, and the two's complement pipelined module is increased from 14.6 MHz to 47.6 MHz. Thus the ones' complement always has better performance on operation clock rate and silicon area to implement multiply-add module. One interesting fact to note in Table 4.1 is that the silicon size required for the four-stage pipelined module is larger than the five-stage module in two's complement as well as that the silicon size required for the three-stage pipelined module is larger than the four-stage pipelined module. This is due to the fact that the designer set different locations of input and output signals. Therefore, the different locations of signals have different routing. It is doubtful that additional designer attention to placement would result in significant reductions in any of the areas.

B. CONCLUSION AND REMARK

This thesis has described the application of a silicon compiler to the design of the basic module of IIR notch filter. The Genesil Silicon Compiler system is a rapid and efficient stand-alone CAD tool for translating an algorithm to a hardware implementation and the subsequent verification. Rapid iterative design, simulation, and timing analysis is possible. It has permitted evaluation of several alternative designs for pipeline multiply-add unit. As a result of this study, it is concluded that the ones' complement number system is better to be used for designing the pipeline IIR notch filter. In

addition, final chip designs are available for one- through five-stages pipeline multiply-add units.

The following are recommendations for further study:

1. To build a faster notch filter chip: We must split the multiplier and the adder into pipeline stages because the fastest operational clock rate is still limited by the multiplier and adder generated by the silicon compiler.

2. To incorporate design for testability at the input and output of the notch filter: It is investigating to see if we can incorporate use of built-in self-test (BIST) for the testability design.

REFERENCES

1. Loomis, H. H., Jr. and Sinha, B., "High-Speed Recursive Digital Filter Realization", Circuits, Systems and Signal Processing, Vol. 3, No. 3, pp. 267-294, 1984.

2. Jangsri, V., Infinite Impulse Response Notch Filter, Master's Thesis, Naval Postgraduate School, Monterey, CA., December 1988.

3. Stanley, W. D., Electronic Communications Systems, Prentice-Hall, Englewood Cliffs, NJ, 1982.

4. Settle, R. H., Design Methodology Using the Genesil Silicon Complier, Master's Thesis, Naval Postgraduate School, Monterey, CA., September 1988.

5. Rockey, R. R., Silicon Complier Implementation of a Kalman Filter Algorithm as an ASIC, Master's Thesis, Naval Postgraduate School, Monterey, CA., December 1988.

6. Davidson, J.C., Implementation of a Design for Testability Strategy Using the Genesil Silicon Compiler, Master's Thesis, Naval Postgraduate School, Monterey, CA., March 1989.

7. J. Wakerly, Digital Design Principles and Practices, Prentice-Hall, Englewood Cliffs, NJ, 1990.

8. N. Weste and K. Eshraghian, Introduction to CMOS VLSI Designs, Addison-Wesley, NY, 1985.

9. Genesil System Compiler Library Volumn III Random Logic Module, Silicon Compiler Corp., 1986.

10. Loomis, H. H., Jr. The Maximum Rate Accumulator, IEEE Tec. EC-15, 4 (1966), 628-639.

11. Chen, T. C., Overlap and Pipeline Processing, Introduction to Computer Architecture, H. Stone, editor, Science Research Associate, Chicago, Chapter 9.

12. Kogge, P. M., The Architecture of Pipelined Computers, Hemisphere Publishing, 1981.

13. Tsui, F.F., LSI/VLSI Testability Design, McGraw-Hill Book Company, 1987.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 54 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
4.	Department Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
5.	Professor Chyan Yang, Code 62YA Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	5
6.	Professor Herschel H. Loomis, Jr., Code 62LM Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	5

7. Professor Dan C. Boger, Code 54BO 2
Department of Administrative Sciences
Naval Postgraduate School
Monterey, California 93943-5000

8. Chinese Naval Academy Library 2
Kaoshiung Tsoying P.O. Box 8494
Taiwan, Republic of China

9. C.C.I.T. Library 1
P.O. Box 335 Ta-Hsi, Tao-Yuan
Taiwan, Republic of China

10. M.E.T.S. Library 1
P.O. Box 90502 Nan-Kan, Taipei
Taiwan, Republic of China

11. T.F.P.G. Library 1
P.O. Box 8761 Ta-Fu, I-Lan
Taiwan, Republic of China

12. Kung, Chih-fu 2
Kaoshiung Tsoying
635 Tsoying-Ta Road
Taiwan, Republic of China